4-2018

Project Phenom: A Smart Bike Project

Stephen Fickas
University of Oregon

NITC
NATIONAL INSTITUTE for TRANSPORTATION and COMMUNITIES

FINAL REPORT

Project Phenom: A Smart Bike Project

NITC-ED-1072 ■ April 2018

NITC is a U.S. Department of Transportation national university transportation center.

TREC
Transportation Insight for Vibrant Communities

PROJECT PHENOM: A SMART BIKE PROJECT

Final Report

NITC-ED-1072

by

Stephen Fickas
University of Oregon

for

National Institute for Transportation and Communities (NITC)
P.O. Box 751
Portland, OR 97207

April 2018

Technical Report Documentation Page

1. Report No. NITC-ED-1072	2. Government Accession No.	3. Recipient's Catalog No.
4. Title and Subtitle Project Phenom: A Smart Bike Project		5. Report Date April 2018
		6. Performing Organization Code
7. Author(s) Stephen Fickas		8. Performing Organization Report No.
9. Performing Organization Name and Address Computer and Information Science Department Deschutes Hall University of Oregon		10. Work Unit No. (TRAIS)
		11. Contract or Grant No.
12. Sponsoring Agency Name and Address National Institute for Transportation and Communities (NITC) P.O. Box 751 Portland, OR 97207		13. Type of Report and Period Covered
		14. Sponsoring Agency Code
15. Supplementary Notes		

16. Abstract

This project introduces students to a hands-on experience building smart-transportation hardware and software. After completing the project, students will gain practical experience with integrating a set of inexpensive, commercial computer components to deliver a ready-to-install system that can extend the transportation grid of a city. They will also learn how to program the system to give bike riders and pedestrians in their community new power. Specifically, the system they build can give bicyclists and pedestrians a virtual push-button, allowing them to place a call for a green light prior to reaching an intersection, thus giving them the actual green light when they reach the intersection.

The project consists of a set of video-based lessons accessible on YouTube. A student is guided through first the steps needed to build the hardware system and then the steps to program the system. By the end of the lessons, a student will have gained valuable skills:

1. An understanding of an *embedded computer* and how it can be the heart of an *Internet of Things* (IoT) system.
2. An introduction to basic electronics and *breadboarding*. (No soldering involved.)
3. Writing simple programs to **test** breadboard wiring: the concept of software being used to test hardware.
4. An introduction to the *relay* concept and the ability it provides to control other machinery.
5. Writing simple programs to test a relay.
6. A basic introduction to *cloud computing* and how it can be used to control the embedded computer.
7. More advanced *programming in C++* to give the system the smarts necessary to interact with bicyclists and pedestrians.
8. *Containerization*. The ability to integrate a set of hardware components into a container that is ready to be installed.

These skills are at the heart of the coming wave of IoT systems.

It is important to note that the actual installation of the system a student builds will require cooperation from the local traffic office. Please contact the author, Stephen Fickas (fickas@cs.uoregon.edu), for assistance in gaining this cooperation, or for other IoT project ideas that do not require city installation.

17. Key Words Smart Transportation, Active Transportation, V2X, IoT	18. Distribution Statement No restrictions. Copies available from NITC: http://nitc.trec.pdx.edu		
19. Security Classification (of this report) Unclassified	20. Security Classification (of this page) Unclassified	21. No. of Pages 20	22. Price

ACKNOWLEDGMENTS

This course was developed with financial support from the National Institute of Transportation and Communities under grant number 1072.

I am grateful to the participants of the South Eugene Robotics Team and, in particular, Dani White.

DISCLAIMER

CITATION

Fickas, Stephen. *Project Phenom: A Smart Bike Project*. NITC-ED-1072. Portland, OR: Transportation Research and Education Center (TREC), 2018.

TABLE OF CONTENTS

LIST OF FIGURES

EXECUTIVE SUMMARY

This project introduces students to hands-on, experience-building, smart-transportation hardware and software. After completing the project, students will gain practical experience with integrating a set of inexpensive, commercial computer components to deliver a ready-to-install system that can extend the transportation grid of a city. They will also learn how to program the system to give bike riders and pedestrians in their community new power. Specifically, the system they build can give bicyclists and pedestrians a virtual push-button, allowing them to place a call for a green light prior to reaching an intersection, thus giving them the actual green light when they reach the intersection.

The project consists of a set of video-based lessons accessible on YouTube. A student is guided through first the steps needed to build the hardware system and then the steps to program the system. By the end of the lessons, a student will have gained valuable skills:

1. An understanding of an *embedded computer* and how it can be the heart of an *Internet of Things* (IoT) system.
2. An introduction to basic electronics and *breadboarding*. (No soldering involved.)
3. Writing simple programs to **test** breadboard wiring: the concept of software being used to test hardware.
4. An introduction to the *relay* concept and the ability it provides to control other machinery.
5. Writing simple programs to test a relay.
6. A basic introduction to *cloud computing* and how it can be used to control the embedded computer.
7. More advanced *programming in C++* to give the system the smarts necessary to interact with bicyclists and pedestrians.
8. *Containerization*. The ability to integrate a set of hardware components into a container that is ready to be installed.

These skills are at the heart of the coming wave of IoT systems.

It is important to note that the actual installation of the system a student builds will require cooperation from the local traffic office. Please contact the author, Stephen Fickas (fickas@cs.uoregon.edu), for assistance in gaining this cooperation, or for other IoT project ideas that do not require city installation.

1.0 INTRODUCTION

In the rush to develop and install smart-transportation (V2X) systems so that they can support connected and autonomous vehicles, two separate and overlapping stakeholders are being left out: bicycles and youth. At best, bicycles are viewed as obstacles to be avoided by sensored-up cars. And in the vision of connected vehicles, children are rarely thought of in any way other than as units to be chauffeured by car. This is an extremely limited view of the future and not the path to a healthy and vibrant community. We offer an alternative path where people on bikes are true partners in the connected V2X space.

These sets of lessons are part of a larger project to create and test a low-cost, and hopefully ubiquitous, system that brings walking and cycling into the smart and connected communities' framework. We are focusing on four key tasks: (1) Develop transportation scenarios for the challenges that children face when biking to school and evaluate those scenarios in a bicycle simulation lab. (2) Develop technology that will address those challenges by tapping into the larger smart-transportation infrastructure through an inexpensive active transportation device (Bike Connect) and, at the same time, extend that infrastructure to open up a new world of bike-friendly features (through the Bike Connect Device). (3) Tie scenarios, simulation and technology development together in a new agile framework especially suited for domains where field tests are problematic. (4) Demonstrate the suitability of our approach by focusing on four schools that are diverse both in the student body and the geographical challenges they present. Our *Project Phenom: A Smart Bike Project*, addresses task 2. It focuses on a bike box that can be installed at existing signal-controlled intersections through simple connections to the controller. Once installed, the bike box can allow a bike rider to use an app to push a virtual call button at the right distance from the signal, giving the rider a green light just in time as s/he approaches the intersection. The bike app is being developed in a separate NITC project, *V2X: Bringing Bikes Into The Mix*.

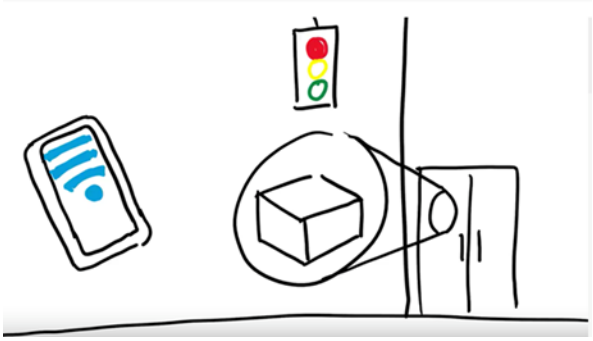


Figure 1: This module allows students to engage with V2X technology by constructing a bike box that, when installed at a street light, can be used with a bike app to trigger a green light from a distance.

The innovation of the Phenom project is that it recognizes that V2X technology does not have to be hidden behind company walls or reserved for only researchers at universities. Instead, developing V2X technology can be made an open project available to anyone, and in particular, students wishing to learn more about the Internet of Things and transportation. These sets of video lessons provide a clear and detailed roadmap on giving students a chance to explore V2X technology and, in the end, produce something that can be used in their own community.

The course topics include: (1) Doing electronic (solderless) breadboarding to connect a modern Internet of Things device, a cellular embedded computer, into the heart of their system. (2) Demonstrating how the embedded computer (a particle electron) can easily control relays, which provide the virtual push-button. (3) Packaging up their system into a container that can be placed on a signal pole. (4)

Lessons on how to employ C++ and object-oriented programming to control the electron from the cloud. When completed, the student will have built a powerful V2X system that is ready to be employed at a signal in their community.

The bicycle is humanity's most energy-efficient invention, and also happens to be space efficient, economically equitable, available to those too young to operate a motorized vehicle, can work within the more decentralized and sprawling landscapes of most American communities better than transit and walking ever can, and happens to be a mode that cities of all sizes are investing in through bikeshare systems. Our work is to utilize new smart- and connected-community thinking to bring the bicycle and youth centrally into the broader area of work so that we are not just working on technology looking for an application but are focused on real societal outcomes and fitting new technology to help meet the social, economic, and environmental challenges of our time.

2.0 FOR EDUCATORS

While the lessons developed by the project are freely available on YouTube, there is a cost to taking on the project in your own class. In this section we provide information that will be useful in making an adoption decision.

2.1 MATERIAL LIST & COURSE BUDGET

- Electron board: $65 (plus $2.99/month for cellular connection)
- Relay: $10 each
- Terminal strip: $10
- Power converter: $35
- Wiring: $5
- Acrylic mounting boards: $10
- Container: $100

The container is the most expensive component. It is not needed until the system is ready to be installed at an intersection. All items and links to their sales page are here: http://www.fasttraq.bike/main/builder/.

2.2 TIME FRAME NEEDED

For the basic hardware setup and testing: five hours. For the more advanced programming, the time needed depends on the students' proficiency with (a) programming in general, and (b) C++ in particular. A student with no prior programming experience will need an intro to C++. Prior to starting our lessons, we recommend a free online course at Udemy: https://www.udemy.com/free-learn-c-tutorial-beginners/. Note that this course contains 18 hours of video. However, we believe it is worth it given that C++ is at the center of many Internet of Things projects; what they learn in the Udemy course will have applicability beyond our specific project.

For a student with knowledge in another language, then a C++ object-oriented programming brush up will be needed. If the student knows Java, in particular, this free guide offers a succinct mapping of Java concepts to C++ concepts: http://www.horstmann.com/ccj2/ccjapp3.html. We estimate that a student who knows Java can follow our lessons in C++ with this site opened as a help guide – it is not necessary to take an entire course on C++ if an object-oriented language is known by the student.

2.3 TARGET AUDIENCE/STUDENT GROUP

The lessons from 1-6 can be taken on by students with some beginning knowledge of programming.

2.4 METHODS

This is very much a project-oriented, hands-on course. The expectation is that the student team will be following the lessons along with the actual hardware on their workbench.

2.5 POTENTIAL PITFALLS OF ACTIVITY OR COMMON MISTAKES STUDENTS MAY MAKE

For any electronics topic, sloppy direction-following will lead to errors. The video lessons hammer this home and provide the points where a second evaluator (the teacher or another student) should be asked to check the work so far. In our experience, students who skip these checkpoints are ones who are in a hurry to finish and can be sloppy both in wiring and programming. We suggest that the teacher insist on checkpoints being maintained.

2.6 UNIQUE CONSIDERATIONS

The finished boxes are ready to be connected to a signal controller. This will require the cooperation of city officials, in particular, staff at the transportation office. However, the boxes can be used for other IoT projects around the school; see section 4.0. Contact Stephen Fickas (fickas@cs.uoregon.edu) for further details.

2.7 ADDITIONAL BACKGROUND MATERIAL

The following references can be used to provide a bigger context for the lessons. For instance, if the teacher wishes to bring in discussion of urban planning as it relates to the connected and autonomous vehicle future, these can be valuable.

Cycling to work in 90 large American cities: new evidence on the role of bike paths and lanes. *Transportation*, *39*(2), 409-432.

Dill, J., & Carr, T. (2003). Bicycle commuting and facilities in major US cities: if you build them, commuters will use them. *Transportation Research Record: Journal of the Transportation Research Board*, (1828), 116-123.

Dill, J., & McNeil, N. (2013). Four types of cyclists? Examination of typology for better understanding of bicycling behavior and potential. *Transportation Research Record: Journal of the Transportation Research Board*, (2387), 129-138.

Doyle, S., Kelly-Schwartz, A., Schlossberg, M., & Stockard, J. (2006). Active community environments and health: the relationship of walkable and safe communities to individual health. Journal of the American Planning Association, 72(1), 19-31.

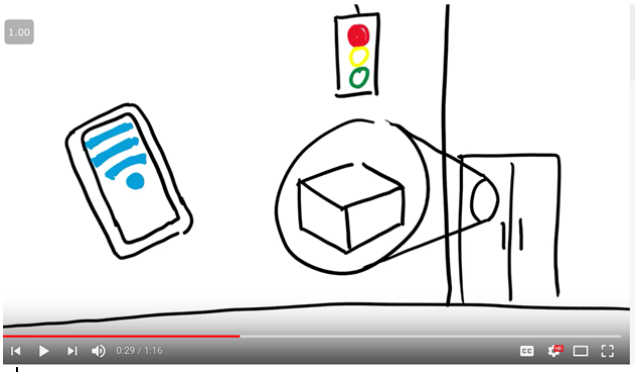
Evers, C., Boles, S., Johnson-Shelton, D., Schlossberg, M., & Richey, D. (2014). Parent safety perceptions of child walking routes. Journal of transport & health, 1(2), 108-115.

Pucher, J., & Buehler, R. (2008). Making cycling irresistible: lessons from the Netherlands, Denmark and Germany. *Transport reviews*, *28*(4), 495-528.

Pucher, J., & Buehler, R. (2016). Safer Cycling Through Improved Infrastructure. American *Journal of Public Health*, 106(12), 2089–2091. http://doi.org/10.2105/AJPH.2016.303507

3.0 LESSON OUTLINE AND CONTENT

The course lessons consist of seven lessons that are hosted on YouTube and are free of charge. As a student proceeds through them, the assumption is that the student has the hardware and software tools necessary to follow along.

Lesson Content	Online Material
3.1 LESSON 1 This lesson provides an overview of the project to give context for the following lessons. It also introduces the two researchers on the project (Figure 2). **Time requirement**: Five minutes. However, this is a point where a teacher can start a larger conversation about the new world of smart transportation and the place of bicyclists and pedestrians. The reading in Section 2.7 should be a good starting point.	 Figure 2: The researchers on the project are introduced in the first lesson. URL: https://youtu.be/28gHZIIchRI
3.2 LESSON 2 This lesson introduces the specific device students will be building, the bike box. It illustrates the communication links between the bike box and a bike rider's app (Figure 3). **Time requirement**: Five minutes. Can be used with Lesson 1 to start a more general discussion of smart transportation.	 Figure 3: This lesson provides a dynamic explanation of the V2X concept students will explore in this module. URL: https://youtu.be/GoglAme_UIc

3.3 LESSON 3

This lesson begins the actual workbench steps for building a bike box.

Time requirement: Two hours.

3.3.1 Particle electron and cloud

The particle electron board is introduced as the heart of the bike box (Figure 6).

3.3.2 Setup instructions

We describe the setup steps for activating the cellular connection that is necessary to give the electron access to the outside world (Figure 4).

Figure 4: Account interface to set up a cellular connection.

URL: https://youtu.be/7Qr0c2-fEhM

3.3.3 Cellular cost of data usage

Because we are using cellular data for communication, there is a charge. We provide information on the cost to transfer cellular data from the electron. Note that the minimum cost (currently $2.99/month) is plenty for our uses (Figure 5).

This part of the lesson also shows how to set a hard limit on your monthly costs and how to deactivate the electron so you are no longer being charged.

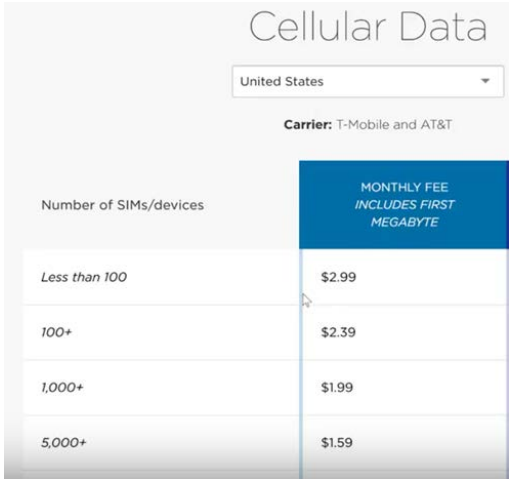


Figure 5: Cell cost options to use transfer cellular data from the electron.

3.3.4 Electron setup

We now turn to actually working with the electron on a workbench (Figure 6).

The following development steps are illustrated in this lesson:

1. Sim card installation
2. Breadboard setup
3. Particle command line interface
4. Set environment variable
5. Login to electron
6. Checking cell signal strength (coding step)
7. Loading your code onto the electron

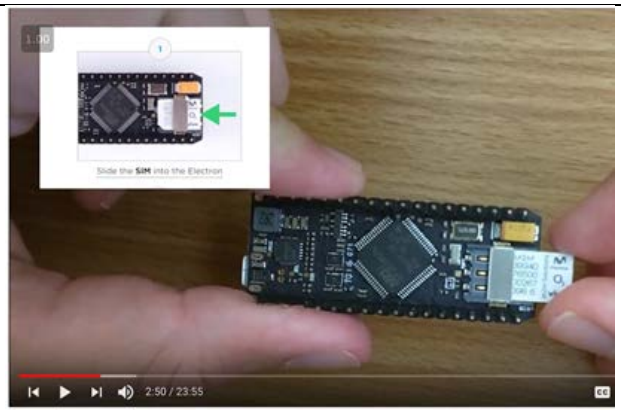


Figure 6: Sim card on electron.

3.3.5 Starting to code

We follow the steps to generate a working program on the electron to turn a led on and off (Figure 7). Coding will be in C++.

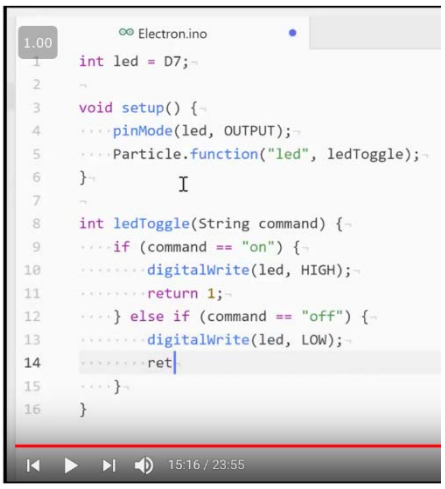


Figure 7: First code for led (in C++).

3.3.6 Sketching breadboard layout

In this part of the lesson, we diagram the way we will connect components. This provides a roadmap to the student of where we are going (Figure 8).

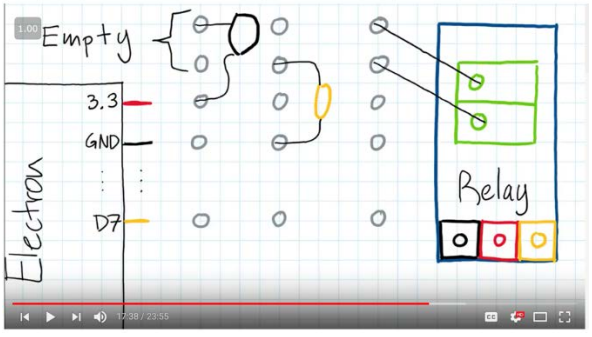


Figure 8: Sketching the layout of the breadboard.

3.3.7 Wiring steps

We illustrate how to follow the diagram we sketched to actually wire components together using a breadboard.

By the end of this lesson, you will have all of your hardware set up and can actually test it (Figure 9).

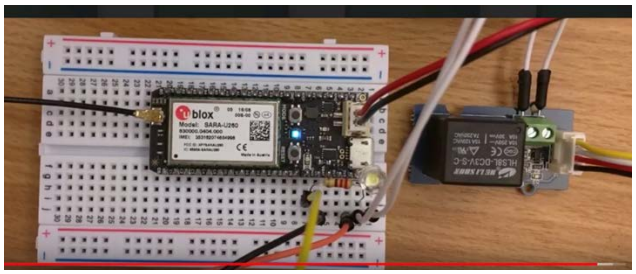


Figure 9: Wiring the breadboard.

3.4 LESSON 4

In this lesson we scale out from one relay (Lesson 3) to four relays.

Time requirement: Three hours

3.4.1 Writing the code

We follow the steps to generalize our code from Lesson 3 to now control four relays instead of one (Figure 10).

3.4.2 Sketching new wiring diagram

Similar to Lesson 3.6, we will first get an overview of what we are doing by creating a diagram as a roadmap.

3.4.3 Adding new hardware

We now introduce the new hardware components we will use. This includes all four relays and a terminal interface that makes wiring simpler (Figure 11).

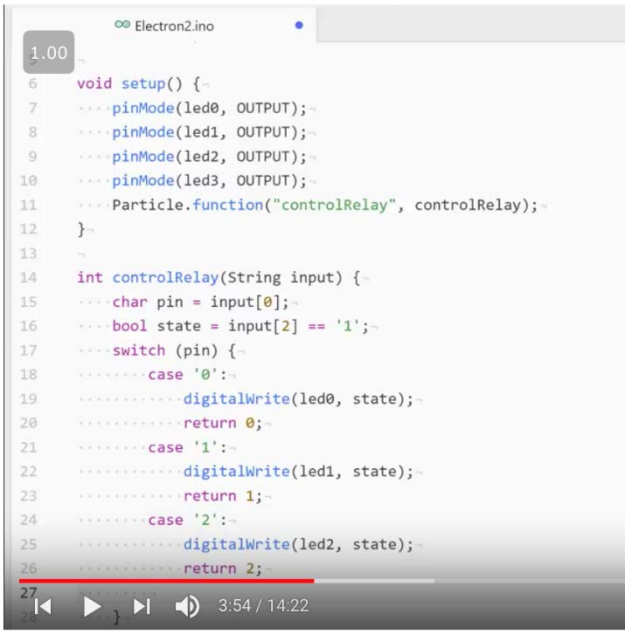


Figure 10: Coding for four relays (in C++).

URL: https://youtu.be/UpWn6LKRmFM

Figure 11: New hardware: Four relays and a terminal interface.

3.4.4 Wiring components

We provide detailed directions for wiring all of the components together, including hints on how to stay organized with so many different wires in play. By the end of this section of the lesson, a student will have the hardware wired and ready to test (Figure 12).

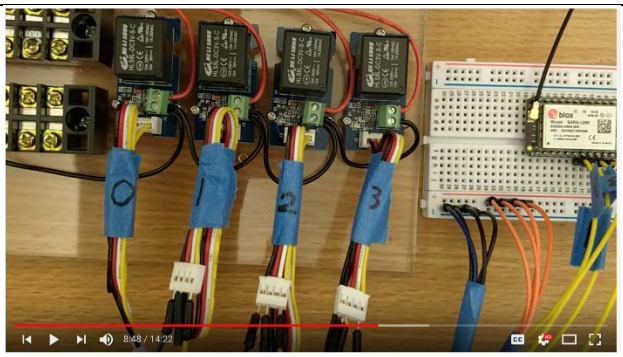


Figure 12: Wiring the components

3.4.5 Testing connections

We demonstrate how to test the connection for each of the four relays by wiring in four leds that are controlled by the four relays.

By the end of Lesson 4, students will be ready to set up the box that will contain the relays.

3.5 LESSON 5

In this lesson, we follow the steps to package everything in a container that can be placed on a traffic pole.

Time requirement: Two hours

3.5.1 Connecting the power converter

The power converter, shown at the top of Figure 13, concerts 120AC to 5DC. Detailed instructions are included on (a) how to wire it and (b) safety precautions to follow.

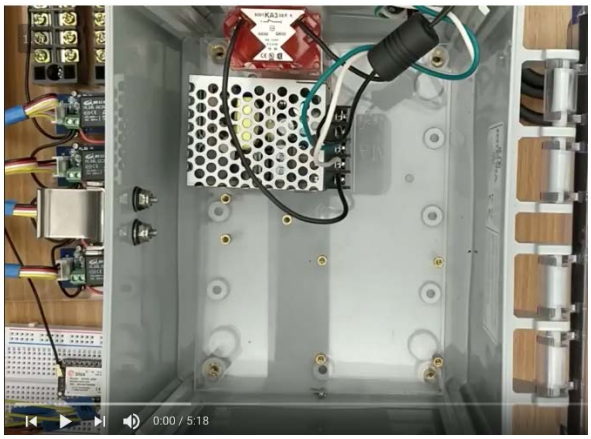


Figure 13: To allow placement on a traffic pole, all components are moved into a container (referred to as containerization).

3.5.2 Connection of relays

The next step is to add the relays to the box and wire up the electron (Figure 14).

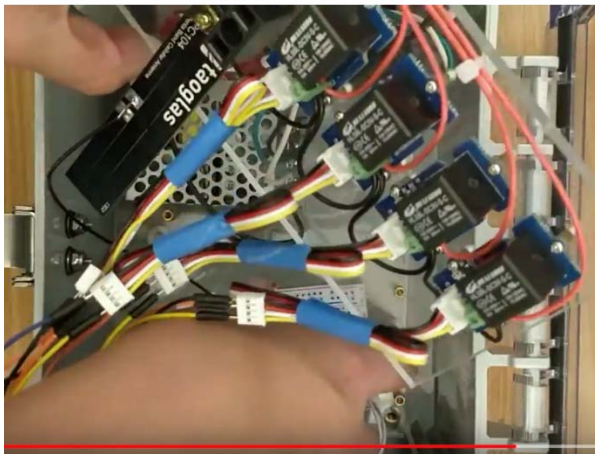


Figure 14: Placing components in the box

URL: https://youtu.be/XwgRSwUQJH0

3.6 LESSON 6 Now that we have a working box, we turn to creating software that will allow us to control it. The big concept here is creating an application programming interface, or API for short. This provides a way for the outside world to both activate the relays as well as monitor the state of the box. Coding is in C++. **Time requirement**: Two hours.	URL: https://youtu.be/QyJVCLI5SmQ

3.7 LESSON 7

In this final lesson, we look over our code one more time. We are looking for ways to make it cleaner. The concept introduced is object-oriented programming (see Figure 15).

Time requirement: Three hours.

3.7.1 Defining classes

We illustrate how the code we have written in previous lessons can now be objectified. Motivation is provided for why using an OOP style makes sense.

3.7.2 Building the Relay class

We create a Relay class that will act as a superclass. It captures the general information about a single relay but does not commit to how relays are controlled.

3.7.3 Building the GPIO_Relay class

We create a new class, GPIO_Relay, that is a subclass of Relay. The GPIO_Relay class provides details on how to control the relays using the GPIO pins on the electron.

3.7.4 Linking with the ino file

We have now completed our OOP development in cpp files. We are ready to link that into the ino file (i.e., the code running on the electron).

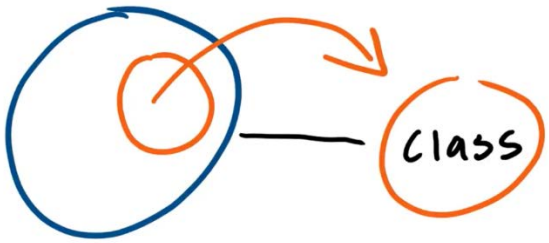


Figure 15: Object-oriented programming

URL: https://youtu.be/opuCTcAqCo0

3.7.5 Testing our code on the electron

The final step is to flash our code to the electron and then call our functions and watch the behavior of the leds and relays on the electron board (Figure 16).

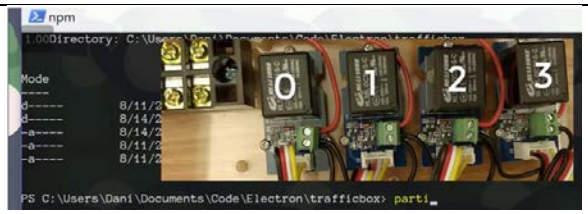


Figure 16: Testing components of the bike box.

4.0 FEEDBACK WANTED

We welcome your feedback on the course. In particular, we make certain assumptions about your class and students:

1. You can afford to purchase the hardware and wiring. Is that a problem?
2. Your students have some programming experience in a non-graphics programming language like Python, Java or C++. Is this a valid assumption?
3. Your students will have access to a laptop or desktop computer that is next to their work area. Is this a valid assumption?

Would you be interested in content material around smart transportation, autonomous vehicles and biking? This might allow you to create a whole unit around these topics with box building as the final project.

We have other projects using the box that do not involve installing it on a signal. In its most general form, the box is a means to turn things off and on remotely. And it can be controlled by any computer with access to the internet. This is shown in detail in videos 3, 4 and 7 where a laptop computer turns on and off leds and relays over the internet. With this setup, you can command the relay from your laptop to control any electrical device (e.g., lights, coffee makers, etc.). You can also look on the web for relay-based projects (e.g., https://www.hackster.io/projects/tags/relay).

Please contact Stephen Fickas (fickas@cs.uoregon.edu) with feedback or questions.

NITC
NATIONAL INSTITUTE for TRANSPORTATION and COMMUNITIES
Transportation Research and Education Center
Portland State University
1900 S.W. Fourth Ave., Suite 175
Portland, OR 97201